EUR 3559 e

EUROPEAN ATOMIC ENERGY COMMUNITY - EURATOM

AN APACHE "SUBROUTINE" FOR GENERATING PADE'S CIRCUITS

by

C. BONA

1967

Joint Nuclear Research Center
Ispra Establishment - Italy
Scientific Information Processing Center - CETIS

This report is on sale at the addresses listed on cover page 4

at the price of FF 4.–	FB 40.–	DM 3.20	Lit. 500	Fl. 3.–

When ordering, please quote the EUR number and the title, which are indicated on the cover of each report.

Printed by SMEETS
Brussels, August 1967

EUR 3559 e

AN APACHE "SUBROUTINE" FOR GENERATING PADE'S CIRCUITS by C. BONA

European Atomic Energy Community - EURATOM
Joint Nuclear Research Center - Ispra Establishment (Italy)
Scientific Information Processing Center - CETIS
Brussels, August 1967 - 22 Pages - 8 Figures - FB 40

With APACHE it is possible to write programs corresponding to analog circuits of common use which can be considered as the "Subroutines" or "Functions" in digital programs.

In fact, when a problem requires a circuit for which a subroutine has been written one can avoid to write the corresponding equations just

EUR 3559 e

AN APACHE "SUBROUTINE" FOR GENERATING PADE'S CIRCUITS by C. BONA

European Atomic Energy Community - EURATOM
Joint Nuclear Research Center - Ispra Establishment (Italy)
Scientific Information Processing Center - CETIS
Brussels, August 1967 - 22 Pages - 8 Figures - FB 40

With APACHE it is possible to write programs corresponding to analog circuits of common use which can be considered as the "Subroutines" or "Functions" in digital programs.

In fact, when a problem requires a circuit for which a subroutine has been written one can avoid to write the corresponding equations just

by adding the deck constituting the subroutine to the APACHE deck representing the problem.

In the following, two subroutines for generating any number of Pade's circuits for fixed and variable time delays are described.

by adding the deck constituting the subroutine to the APACHE deck representing the problem.

In the following, two subroutines for generating any number of Pade's circuits for fixed and variable time delays are described.

EUR 3559 e

EUROPEAN ATOMIC ENERGY COMMUNITY - EURATOM

AN APACHE "SUBROUTINE" FOR GENERATING PADE'S CIRCUITS

by

C. BONA

1967

Joint Nuclear Research Center
Ispra Establishment - Italy

Scientific Information Processing Center - CETIS

SUMMARY

With APACHE it is possible to write programs corresponding to analog circuits of common use which can be considered as the "Subroutines" or "Functions" in digital programs.

In fact, when a problem requires a circuit for which a subroutine has been written one can avoid to write the corresponding equations just by adding the deck constituting the subroutine to the APACHE deck representing the problem.

In the following, two subroutines for generating any number of Pade's circuits for fixed and variable time delays are described.

KEYWORDS

PROGRAMMING,
COMPUTERS,
ANALOG SYSTEMS,
DIGITAL SYSTEMS,
CIRCUITS.

Codes

An APACHE "Subroutine" for Generating Pade's circuits (+)

INTRODUCTION

When, in the framework of a larger job, a circuit to simulate a fixed or variable delay is needed, a lot of tedious work has to be done to implement one of the many circuits which approximate the delay operator (Pade approximation, Bode approximation, etc.)

In fact there are no conceptual difficulties, as these circuits are well defined and very efficient formulas are provided to calculate the order of approximation needed based on the maximum frequency to be delayed and on the length of the delay.

The only difficulty which may arise is the choice of the scale factors to be given to each intermediate variable, the other things that one has to do being routine works like computing the pot setting values and drawing the circuit etc.

With APACHE (foot note 1) all these routine works are avoided, being automatically done by the compiler, while a new problem arises: the given transfer functions must be transformed in a system of differential equations of first order; besides an estimate must be given of the maximum values of all variables.

On the other side with APACHE the program corresponding to a given circuit can be written once for ever, punched on cards and kept for every time it is needed.

But there is still something more, with APACHE one need not to write for each order of approximation the

(+) Manuscript received on May 22, 1967.

appropriate program, it is possible to write the program in such a way to generate as many different delay circuits as needed , and each of them of the desired order.

This has been done for the Pade circuit both for fixed and variable time delays.

The aim of this work is to present these two APACHE "Subroutines" (foot note 1) and to give an idea of how they have been written.

How to use the programs

The procedure for using these programs, which consist of two decks of approximately 50 cards each, is very simple:

1) Add the deck (or the decks) to the deck of cards representing your problem
2) Add the following statements: (only those requested by the type of delay you need in your problem)

a) For fixed time delays

EQUATIONS

FF(i) = name of the i^{th} variable to be delayed

Name of the i^{th} variable delayed = GF(i)

PARAMETERS

NRITF = number of delay circuits requested

ORITF(i) = order of the i^{th} delay circuit (up to 20)

TRFIX(i) = value (in sec.) of the i^{th} delay

FFINZ(i) = value of the i^{th} variable to be delayed at t=0

FFMAX(i) = maximum value of the i^{th} variable to be delayed

b) For variable time delays

EQUATIONS

FV(i) = name of the i^{th} variable to be delayed

...

Name of the i^{th} variable delayed = GV(i)

...

PARAMETERS

NRITV = number of delay circuits requested

...

ORITV(i) = order of the i^{th} delay circuit (up to 20)

...

TVINZ(i) = length of the i^{th} delay at t=0

...

TVMIN(i) = maximum length of the i^{th} delay

...

TVMAX(i) = maximum length of the i^{th} delay

...

TVINZ(i) = value of the i^{th} variable to be delayed at t=0

...

FVMAX(i) = maximum value of the i^{th} variable to be delayed

Pade's approximation of $e^{-p\tau}$

As is well known, the transfer function of a pure time delay can be approximated as follows (n^{th} order Pade's approximation):

1) $$G(s)/F(s) = e^{-s\tau} \simeq P_n/Q_n$$

where $G(s) = \mathcal{L}(F(t-\tau))$ and $F(s) = \mathcal{L}(F(t))$ and

$$P_n = h_o - h_1\cdot(s\tau)+\ldots+(-1)^i\cdot h_i\cdot(s\tau)^i+\ldots+(-1)^n\cdot h_n\cdot(s\tau)^n$$

$$Q_n = h_o + h_1\cdot(s\tau)+\ldots\ldots+h_i(s\tau)^i+\ldots\ldots+h_n(s\tau)^n$$

$$h_o = 1,\ h_1 = 1/2,\ldots\ h_1 = \frac{n(n-1)\ldots(n-i+1)}{2n(2n-1)\ldots(2n-i+1).i!},\ldots$$

Note that P_n is quite the same as Q_n except for the sign of the i^{th} terms for odd values of i.

We can write: $G(s).Q_n = F(s).\ P_n$ from which, dividing both terms by $(s\tau)^n$, we obtain:

$$2)\ \frac{h_o}{(s\tau)^n} \cdot (F-G) - \frac{h_1}{(s\tau)^{n-1}} \cdot (F+G)+\ldots+ \frac{(-1)^i . h_i}{(s\tau)^{n-i}} (F-(-1)^i . G)+\ldots+$$

$$+ \frac{(-1)^n . h_n}{(s\tau)^o} \cdot (F-(-1)^n . G) = 0$$

and substituting $k_{n-i} \frac{n!}{(2n)!}$ to h_i

$$3)\ \sum_o^n {}_i \frac{k_i^n}{(s\tau)^i} \left[(-1)^{n-i} . F-G\right] = 0 \qquad k_o=1,\ldots,k_i^n = \frac{(n+i)!}{i!(n-i)!}$$

which is the classical expression from which the Pade's circuit is deduced.

APACHE equations for the Pade's circuit

The easiest way to deduce from eq.3 the Pade's circuit is to rewrite it as follows

$$3')\quad G = (-1)^n . F+Q_1 \quad ; \quad Q_1 = \sum_1^n {}_i \frac{k_i}{(s\tau)^i} \left[(-1)^{n-i} . F-G\right]$$

Thus G can be obtained by adding to Q_1 + F or -F depending on the order (n) of the Pade desired.

Q_1 in its turn is made up by integrating several times terms like (-F-G) and (+F-G) and by summing them up.

Fig.1 shows, in three steps, for clearness' sake, how Q_1 can be obtained by a simple chain of amplifiers. (no sign inversion is considered here)

Note how the gains have been split between stages; this makes them more uniform and much more easy to be computed as

$$\frac{k_i^n}{k_{i-1}^n} = \frac{(n+i)(n-i+1)}{i}$$

Having Q_1, G is immediately obtained by combining it with F and the circuit completed by feeding back F±G.

In fig. 2 both schemes for n odd and even are reported.

The circuit for variable time delays can be deduced in the same way, but it must be remembered that eq.2 is no more rigorous, as well as the splitting of the term $\frac{1}{\tau^n}$ between the different integration stages.

However, experience shows that the circuits in fig.3 give a rather good approximation of a variable time delay provided that the delay (τ) is slowly varying with respect to F.

The APACHE equations for the Pade circuit for fixed and variable delays are easily deduced from fig. 2 and 3 where the circuits for an even and an odd Pade order (N) are reported. (no sign inversion is considered there)

The names between square brackets are the names used in the APACHE program.

Note that, for reasons which will be explained later, the variables which are output from integrators may have two different names depending on the I.C. value of the function to be delayed.

In fig. 4 and 5 the complete listing of the APACHE programs for fixed and variable Pade circuits are reported.

We leave to the readers to go through the equations in order to see how they correspond to the circuits in fig. 2 and 3.

Note how, by means of the APACHE "DO" statement and of the option "NULL", it has been possible to write one program for generating whatever number of Pade circuits of any desired order (up to 20). (Foot note 2)

Variables scaling

No problem arises for scaling variables like GF(k) and GV(k) (fig. 2 and 3) whose maximum value is obviously equal to the maximum value of the variables FF(k) and FV(k) to be delayed, (these values, called FFMAX(k) and FVMAX(k), must be given by the user).

FQFIX(k), GQFIX(k), QNFIX(k,1) which, according to the Pade's order are equal to ±FF(k), -GF(k), ±FF(k)+GF(k), have been scaled assuming a maximum equal to 2 * FFMAX(k), which is conservative.

To scale the intermediate variables QNFIX(k,i) (QNVAR(k,i) and SNVAR(k,i) for variable delays), whose maximum values are not easily predictable, the transfer function between each of them and FF(k) has been calculated and then, looking at their Bode diagrams, it has been found that, for any value of k and i, their amplitude never exceeds the value of 2.2, thus even for these variables a maximum value equal to 2 * FFMAX(k) has been taken.

As an example, the amplitude vs. frequency diagrams of $\mathcal{L}$(QNFIX(k,i))/$\mathcal{L}$(FF(k)) for a Pade of the fourth order have been reported in fig. 6 and 7.

Set of INITIAL CONDITION values for static check

The IC values of the variables of a Pade circuit depend on the values of the function to be delayed and its derivatives τ seconds earlier; we supposed that our Pade circuits will always be used in connection with systems being stationary for $t < 0$, (foot note 3) thus the IC value for the Pade can be easily deduced imposing that all derivative be zero; we have : (fig. 2)

$$Q_i(t=0) = \begin{cases} \text{N EVEN} \begin{cases} 2.F(t=0) & \text{for i EVEN} \\ 0 & \text{for i ODD} \end{cases} \\ \text{N ODD} \begin{cases} 0 & \text{for i EVEN} \\ 2.F(t=0) & \text{for i ODD} \end{cases} \end{cases}$$

(N = Pade's circuit order)

In order to be able to make a complete static check of the circuit, the I.C. conditions which are zero must be changed into some new fictive IC conditions to be taken from the TEST reference.

The problem, which is well known by analogists, is to choose some fictive IC condition which causes the voltages all over the circuit to be different from zero but not greater than 100V and at the same time to minimize the number of potentiometers requested for it.

In fig. 8 the schemes in fig. 2 are repeated writing beside each element the corresponding IC condition for both cases $F(t=0)\neq 0$ and $F(t=0)=0$.

(This figure corresponds to the analog schemes as deduced from the APACHE output listing)

Note the additional terms from the TEST reference which have been added to the summers which compute GF(k), FQFIX(k) and GQFIX(k) to have a complete set of IC $\neq$ 0 and 100V with only two potentiometers. (Foot note 3)

As it is impossible to use the NULL option in connection with the variable list (i.e. to introduce the comment TEST in a variable definition depending on the value of its IC), to cover all possible cases we have been obliged to write three sets of equations, which hold respectively for:

1) F(t=0) = 0

2) F(t=0) $\neq$ 0 N EVEN

3) F(t=0) $\neq$ 0 N ODD

These three sets of equations are all included in the deck, the appropriate set being selected by deleting the other two by means of the NULL option (see fig. 4 and 5 and coefficient CXF(j), GXF(j) and the DO statement DO, j=ORIFT(k),...)

CONCLUSIONS

Our experience has shown that to the computer user a lot of thinking and of tedious work is avoided by means of these routines, therefore we are looking forward to see if it is possible to write APACHE subroutines for other circuits of common use. (e.g. for blocks whose transfer function is assigned).

On the other side, the APACHE language, although it has shown to be versatile enough to cover all possible cases for the Padé circuits, appeared rather cumbersome in use for similar problems, because of the lack of "EQUIVALENCE" statements and of some sort of IF statement (or NULL option) at Parameter and Variable level, which oblige to use fixed names, indexed variables etc. (see above).

Therefore, in the new APACHE version, which is being written now, all these points will be considered.

We hope that these possibilities of organizing the APACHE programs in subroutines will offer still more comfort to the analogue computer user.

Foot note 1

APACHE is a digital program which, starting from a set of equations written in a very intuitive language, performs all the necessary operations to implement them on an analog computer i.e.

a) Reduces the equation in a Standard form corresponding to the characteristic of the analogue elements which will compute them.

b) Performs the scaling of the equations and computes the setting values of potentiometers.

c) Computes the static check value of every element in the circuits.

For further details see : Simulation March 1965 and January 1966 and item 1 in the enclosed bibliography.

An attempt of writing two Apache subroutines like those presented in this paper was done since 1963 by J. Gamp (Bibliography 3 and 4) but only with the last version of the code (July 1965) it has been possible to write them in a really general and usable way.

Obviously, due to the parallel nature of the Apache language, the similarity between APACHE and Fortran subroutines is only formal and consists in the fact that both represent a part of the program that is written once for all.

Thus the user will not find neither CALL or RETURN or similar statements while the physical location of the subroutine deck in the main program is immaterial.

Foot note 2

The APACHE "DO" statement is formally similar to the FORTRAN DO statement but has a different meaning. The APACHE DO is only a means to get greater compactness when many similar equations have to be written (ex. partial differential equations implemented by finite difference method); it does not mean: perform the computation within the DO range for all the indicated values of the indexes, but : consider the equation within the DO range as being written for all the indicated values of the indexes and generate the corresponding circuits.

The NULL option is a means to delete some equations or some part of them: any equation or term which is multiplied by a parameter whose value in the parameter list is 0, NULL is automatically deleted from the problem.

The aim of this option is to give the possibility to include in the range of a DO equations which differ between each other depending on the value of the index.

Foot note 3

The terms ±TREF, which in the Apache listing will appear as an "external" variables, correspond to the test reference voltage; the elements which have an input fed by this variable must be connected to ±TEST REF.

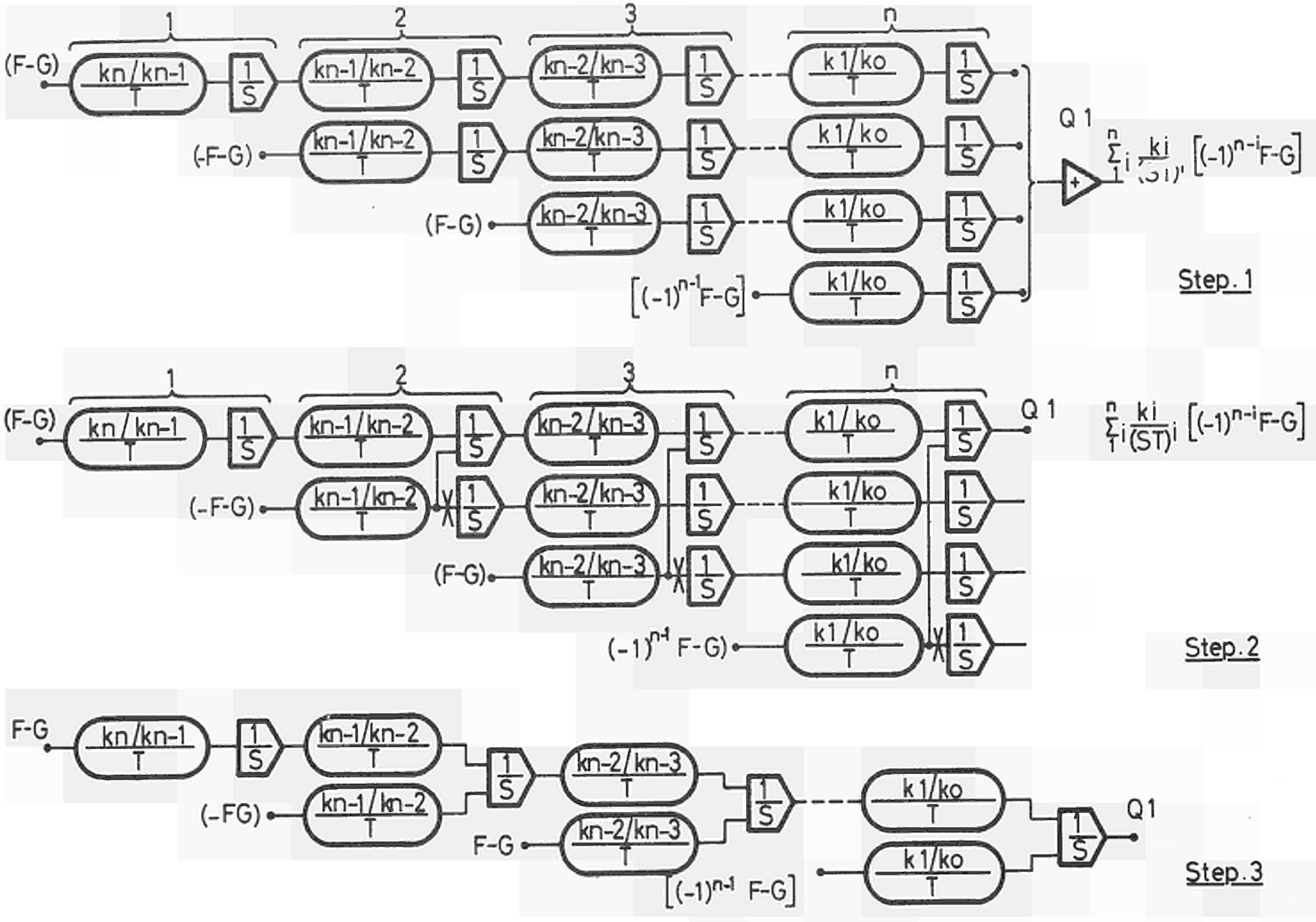


Fig. 1 Implementation of Q1 = $\sum_{1}^{n} i \frac{ki}{(s\tau)^{i}} \left[(-1)^{n-1} \cdot F-G \right]$.

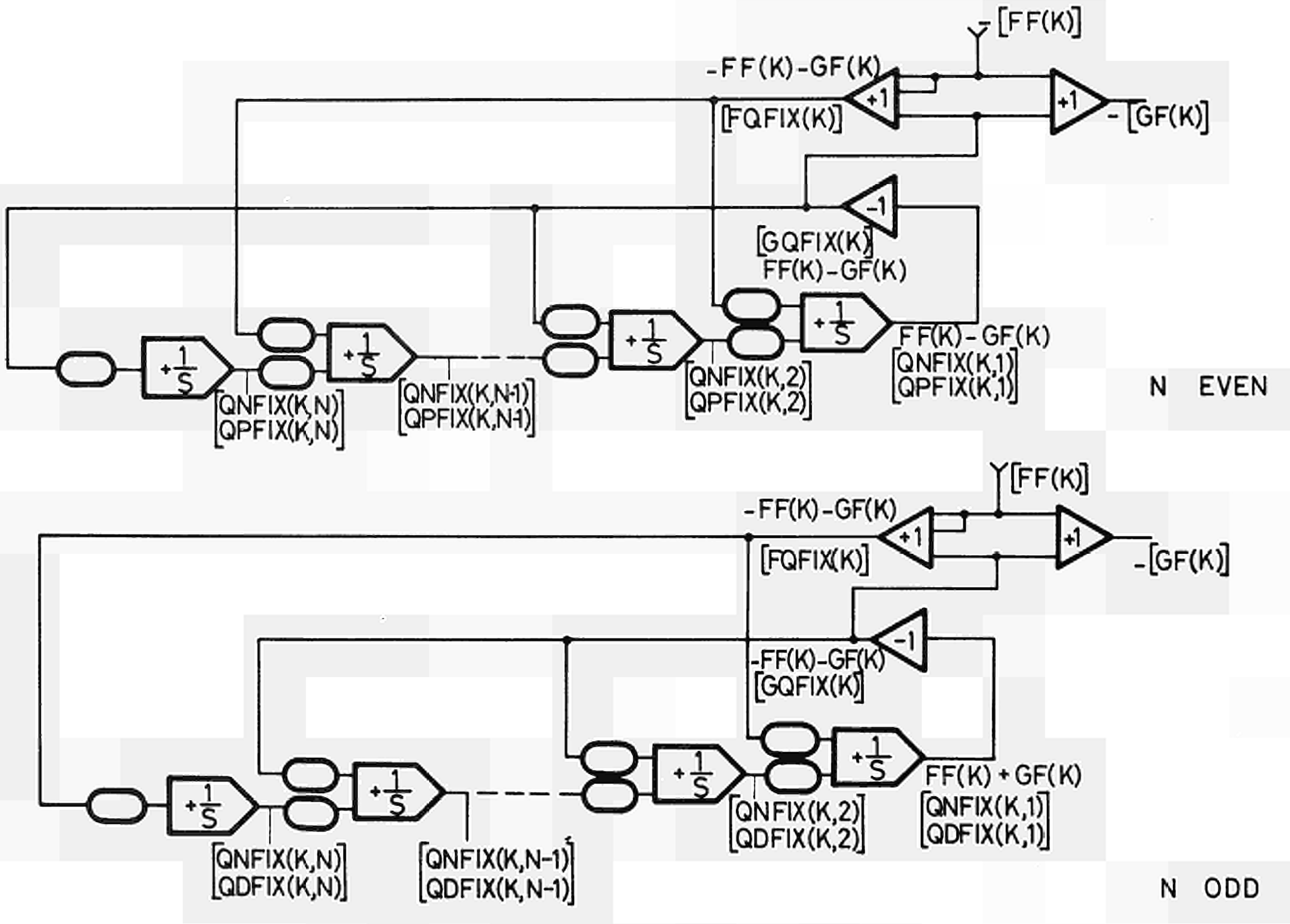


Fig. 2 Pade circuits for fixed time delays.
Odd and even order approximations (N)

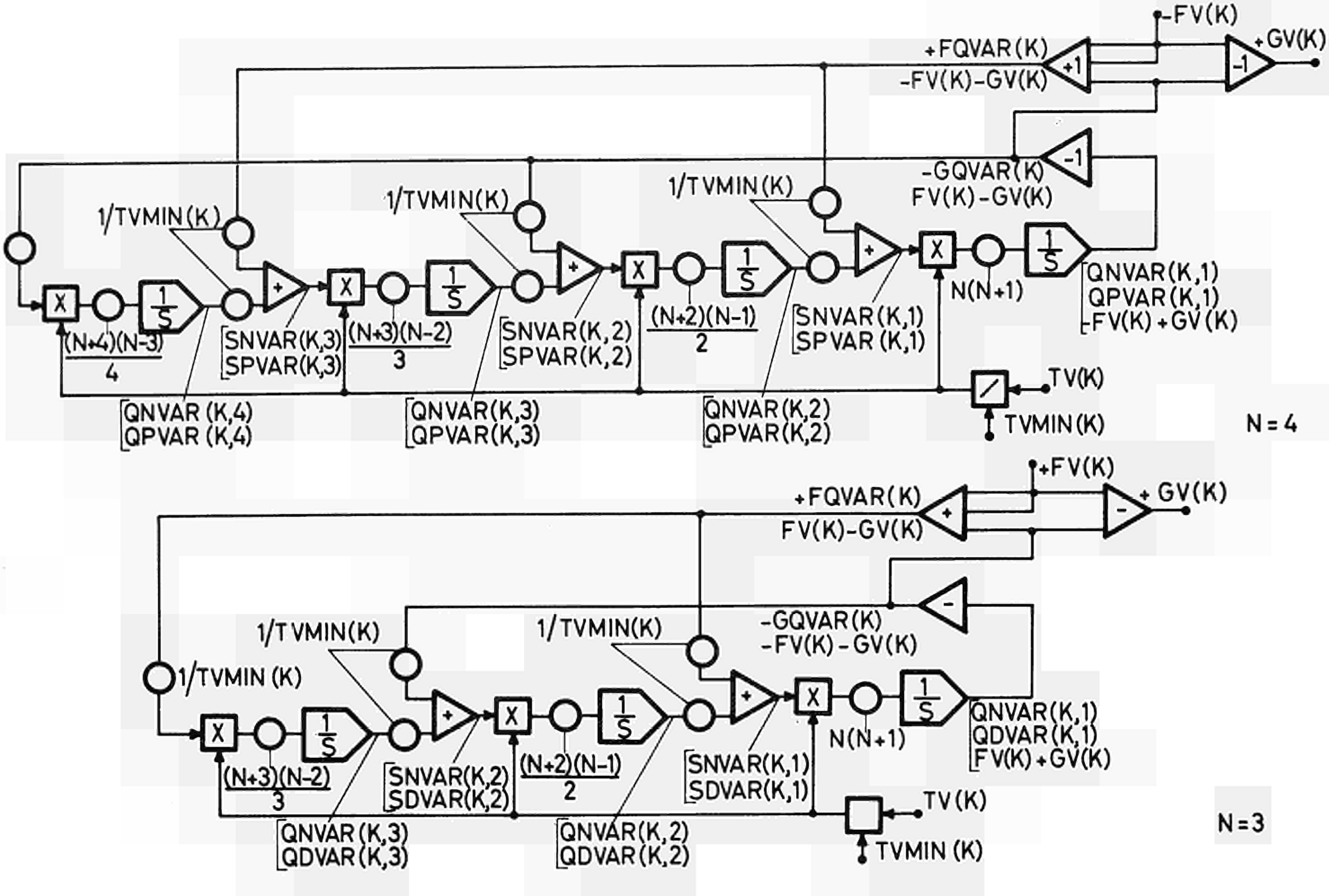


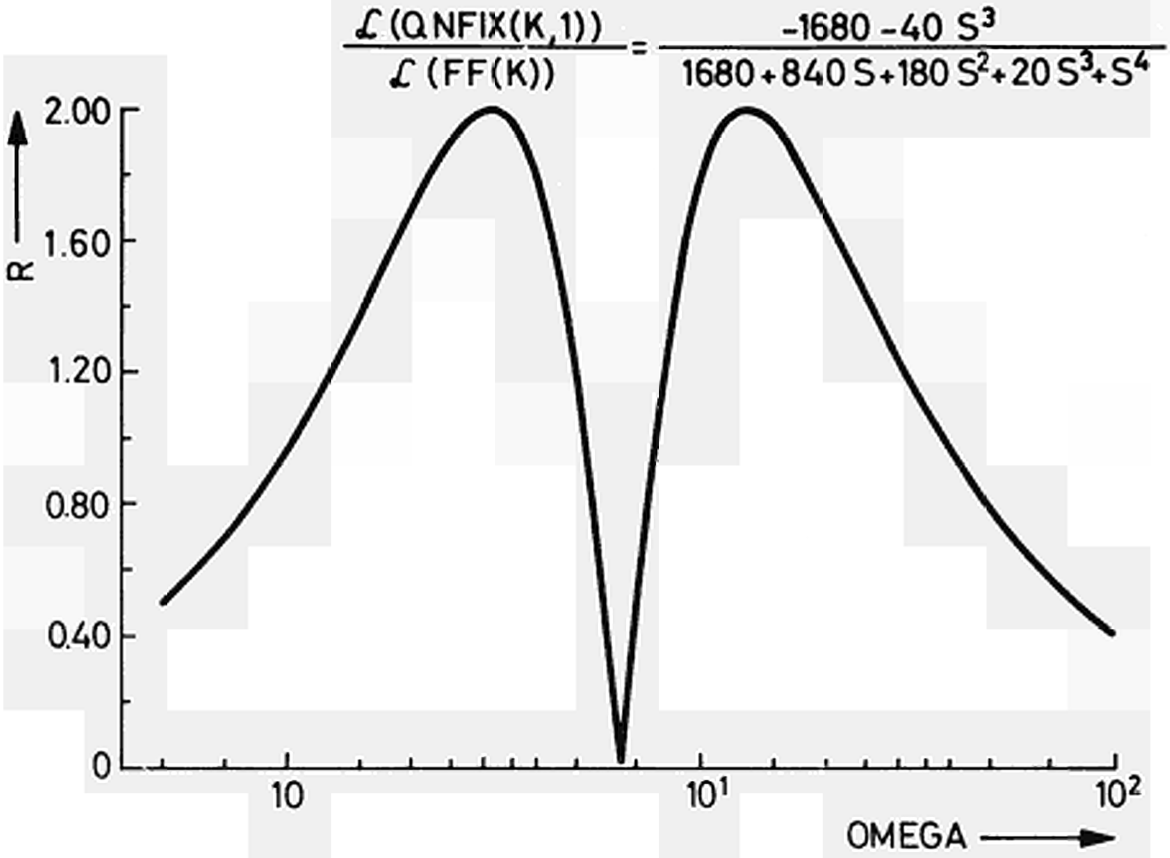
Fig. 3 Pade circuits for variable time delays.
Odd and even order approximation

```
     PARAMETERS                                                       RF001

     DO,9999,I=1,20,2                                                 RF002
     CDISP(I)=1                                                       RF003
9999 CPARI(I)=0,NULL                                                  RF004
     DO,9998,I=2,20,2                                                 RF005
     CDISP(I)=0,NULL                                                  RF006
9998 CPARI(I)=1                                                       RF007
     CXF(0)=1                                                         RF008
     CXF(1)=0,NULL                                                    RF009
     GXF(0)=0,NULL                                                    RF010
     GXF(1)=1                                                         RF011
     DO,9997,K=1,NRITF                                                RF012
     MXMF(K)=ORITF(K)+1                                               RF013
     MXF(K)=ORITF(K)-1                                                RF014
     LXF(K)=(FFINZ(K)-0)/(FFINZ(K)-0)                                 RF015
     PCF(K,1)=ORITF(K)*(ORITF(K)+1)                                   RF016
     DO,9996,I=2,ORITF(K)                                             RF017
9996 PCF(K,I)=(ORITF(K)+I)*(ORITF(K)-I+1)/I                           RF018
9997 QFMAX(K)=2*FFMAX(K)                                              RF019

     VARIABLES                                                        RF020

     TREFF=1,1,TEST                                                   RF020 1
     DO,9995,K=1,NRITF                                                RF021
     FF(K)=FFINZ(K),FFMAX(K),EXACT                                    RF022
        GF(K)=,FFMAX(K),EXACT                                         RF023
     QNFIX(K,1)=0           ,QFMAX(K),EXACT                           RF024
     DO,9994,I=2,ORITF(K)                                             RF025
9994 QNFIX(K,I)=-QFMAX(K),QFMAX(K),EXACT,TEST                         RF026
     QPFIX(K,1)= 0          ,QFMAX(K),EXACT                           RF027
     QPFIX(K,2)=-2*FFINZ(K),QFMAX(K),EXACT                            RF028
     DO,9993,I=4,MXMF(K),2                                            RF029
     QPFIX(K,I-1)=-QFMAX(K),QFMAX(K),EXACT,TEST                       RF030
9993 QPFIX(K,I)=-2*FFINZ(K),QFMAX(K),EXACT                            RF031
     QDFIX(K,1)=-2*FFINZ(K),QFMAX(K),EXACT                            RF032
     DO,9992,I=3,MXMF(K),2                                            RF033
     QDFIX(K,I-1)=-QFMAX(K),QFMAX(K),EXACT,TEST                       RF034
9992 QDFIX(K,I)=-2*FFINZ(K),QFMAX(K),EXACT                            RF035
     GQFIX(K)=,QFMAX(K),EXACT                                         RF035 1
9995 FQFIX(K)=,2*FFMAX(K),EXACT                                       RF036

     EQUATIONS                                                        RF037

     DO,9991,K=1,NRITF                                                RF038
     DO,9991,J=ORITF(K),ORITF(K)                                      RF039
     DO,9991,S=LXF(K),LXF(K)                                          RF040
     FQFIX(K)=2*FF(K)+CXF(S)*(GQFIX(K)+1.8*QFMAX(K)*TREFF)+GXF(S)*CPARIRF041 1
    1(J)*(GQFIX(K)+0.9*QFMAX(K)*TREFF)+GXF(S)*CDISP(J)*(GQFIX(K)+0.9*QFRF041 2
    2MAX(K)*TREFF)                                                    RF041 3
     GQFIX(K)=CXF(S)*(QNFIX(K,1)-0.9*QFMAX(K)*TREFF)+GXF(S)*CPARI(J)*(QRF041 4
    1PFIX(K,1)-0.9*QFMAX(K)*TREFF)+GXF(S)*CDISP(J)*QDFIX(K,1)         RF041 5
     GF(K)=CXF(S)*CPARI(J)*(GQFIX(K)+0.5*QFMAX(K)*TREFF+FF(K))-CXF(S)*CRF042 1
    1DISP(J)*(GQFIX(K)+0.5*QFMAX(K)*TREFF+FF(K))+GXF(S)*CPARI(J)*(GQFIXRF042 2
    2(K)+0.9*QFMAX(K)*TREFF+FF(K))-GXF(S)*CDISP(J)*(GQFIX(K)+FF(K))   RF042 3
     DO,9990,I=1,MXF(K)                                               RF043
9990 CXF(S)*   TF(K)*DER1(QNFIX(K,I))=PCF(K,I)*(-QNFIX(K,I+1)+GQFIX(K) RF044
    1*CPARI(I)-FQFIX(K)*CDISP(I))                                     RF044
     CXF(S)*CPARI(J)*   TF(K)*DER1(QNFIX(K,J))=PCF(K,J)*GQFIX(K)      RF045
     CXF(S)*CDISP(J)*   TF(K)*DER1(QNFIX(K,J))=-PCF(K,J)*FQFIX(K)     RF046
     DO,9989,I=1,MXF(K)                                               RF047
9989 GXF(S)*CPARI(J)*   TF(K)*DER1(QPFIX(K,I))=-PCF(K,I)*(QPFIX(K,I+1)+FRF048
    1QFIX(K)*CDISP(I)-GQFIX(K)   *CPARI(I))                           RF048
     GXF(S)*CPARI(J)*   TF(K)*DER1(QPFIX(K,J))=PCF(K,J)*GQFIX(K)      RF049
     DO,9988,I=1,MXF(K)                                               RF050
9988 GXF(S)*CDISP(J)*   TF(K)*DER1(QDFIX(K,I))=-PCF(K,I)*(QDFIX(K,I+1)-GRF051
    1QFIX(K)   *CPARI(I)+FQFIX(K)*CDISP(I))                           RF051
9991 GXF(S)*CDISP(J)*    TF(K)*DER1(QDFIX(K,J))=-PCF(K,J)*FQFIX(K)    RF052
```

Fig. 4 Listing of the APACHE subroutine for fixed time delays.

```
      PARAMETERS                                                          RV000

      DO,9980,I=1,20,2                                                    RV001
      VDISP(I)=1                                                          RV002
 9980 VPARI(I)=0,NULL                                                     RV003
      DO,9979,I=2,20,2                                                    RV004
      VDISP(I)=0,NULL                                                     RV005
 9979 VPARI(I)=1                                                          RV006
      CXV(0)=1                                                            RV007
      CXV(1)=0,NULL                                                       RV008
      GXV(0)=0,NULL                                                       RV009
      GXV(1)=1                                                            RV010
      DO,9978,K=1,NRITV                                                   RV011
      MXMV(K)=ORITV(K)-1                                                  RV012
      MXVM(K)=ORITV(K)+1                                                  RV013
      LXV(K)=(FVINZ(K)-0)/(FVINZ(K)-0)                                    RV014
      PCOV(K,1)=ORITV(K)*(ORITV(K)+1)                                     RV015
      DO,9977,I=2,ORITV(K)                                                RV016
 9977 PCOV(K,I)=(ORITV(K)+I)*(ORITV(K)-I+1)/I                             RV017
      QVMAX(K)=2*FVMAX(K)                                                 RV018
 9978 SVMAX(K)=2*FVMAX(K)/TVMIN(K)                                        RV019


      VARIABLES                                                           RV020
      TREFV=1,1,TEST                                                      RV020 1
      DO,9976,K=1,NRITV                                                   RV021
      FV(K)=FVINZ(K),FVMAX(K),EXACT                                       RV022
      GV(K)=,FVMAX(K),EXACT                                               RV023
      TV(K)=TVINZ(K),TVMAX(K),EXACT                                       RV024
      MINRV(K)=,TVMAX(K),EXACT                                            RV025
      UDTXV(K)=,1                                                         RV026
      FQVAR(K)=,2*FVMAX(K),EXACT                                          RV027
      GQVAR(K)=,QVMAX(K),EXACT                                            RV027 1
      QNVAR(K,1)=0            ,QVMAX(K),EXACT,TEST                        RV028
      SNVAR(K,1)=,SVMAX(K),EXACT                                          RV029
      DO,9975,I=2,ORITV(K)                                                RV030
      QNVAR(K,I)=QVMAX(K),QVMAX(K),EXACT,TEST                             RV031
 9975 SNVAR(K,I)=,SVMAX(K),EXACT                                          RV032
      QPVAR(K,1)  =0             ,QVMAX(K),EXACT,TEST                     RV033
      SPVAR(K,1)=,SVMAX(K),EXACT                                          RV034
      DO,9974,I=3,MXVM(K),2                                               RV035
      QPVAR(K,I-1)=2*FVINZ(K),QVMAX(K),EXACT                              RV036
      SPVAR(K,I-1)=,SVMAX(K),EXACT                                        RV037
      QPVAR(K,I)=QVMAX(K),QVMAX(K),EXACT,TEST                             RV038
 9974 SPVAR(K,I)=,SVMAX(K),EXACT                                          RV039
      DO,9976,I=2,MXVM(K),2                                               RV040
      QDVAR(K,I-1)=2*FVINZ(K),QVMAX(K),EXACT                              RV041
      SDVAR(K,I-1)=,SVMAX(K),EXACT                                        RV042
      QDVAR(K,I)=QVMAX(K),QVMAX(K),EXACT,TEST                             RV043
 9976 SDVAR(K,I)=,SVMAX(K),EXACT                                          RV044


      EQUATIONS                                                           RV045

      DO,9973,K=1,NRITV                                                   RV046
      DO,9973,J=ORITV(K),ORITV(K)                                         RV047
      DO,9973, S=LXV(K),LXV(K)                                            RV048
      GV(K)=CXV(S)*(GQVAR(K)-0.5*QVMAX(K)*TREFV)+GXV(S)*VPARI(J)*(GQVAR(  RV049
     1K)-0.9*QVMAX(K)*TREFV)+GXV(S)*VDISP(J)*GQVAR(K)+VPARI(J)*FV(K)-VDI  RV050
     2SP(J)*FV(K)                                                         RV050 1
      FQVAR(K)=-CXV(S)*GQVAR(K)-GXV(S)*VPARI(J)*(GQVAR(K)-0.9*QVMAX(K)*T  RV051
     1REFV)-GXV(S)*VDISP(J)*(GQVAR(K)+0.9*QVMAX(K)*TREFV)-VPARI(J)*2*FV(  RV051 1
     2K)+VDISP(J)*2*FV(K)                                                 RV051 2
      GQVAR(K)=CXV(S)*(QNVAR(K,1)+0.9*QVMAX(K)*TREFV)+GXV(S)*VPARI(J)*(Q  RV052
     1PVAR(K,1)+0.9*QVMAX(K)*TREFV)+GXV(S)*VDISP(J)*QDVAR(K,1)            RV052 1
      MINRV(K)=TVMIN(K)                                                   RV053
      UDTXV(K)=MINRV(K)/TV(K)                                             RV054
      DO,9972,I=1,MXMV(K)                                                 RV055
      CXV(S)*DER1(QNVAR(K,I))=PCOV(K,I)*SNVAR(K,I)*UDTXV(K)               RV056
      GXV(S)*VPARI(J)*DER1(QPVAR(K,I))=PCOV(K,I)*SPVAR(K,I)*UDTXV(K)      RV057
      GXV(S)*VDISP(J)*DER1(QDVAR(K,I))=PCOV(K,I)*SDVAR(K,I)*UDTXV(K)      RV058
      CXV(S)*SNVAR(K,I)=QNVAR(K,I+1)/TVMIN(K)-VPARI(I)*VPARI(J)/TVMIN(K)  RV059
     1*GQVAR(K)   +VDISP(I)*VDISP(J)/TVMIN(K)*FQVAR(K)+VDISP(I)*VPARI(J)/  RV060
     2TVMIN(K)*FQVAR(K)-VPARI(I)*VDISP(J)/TVMIN(K)*GQVAR(K)               RV061
      GXV(S)*VPARI(J)*SPVAR(K,I)=QPVAR(K,I+1)/TVMIN(K)-VPARI(I)/TVMIN(K)  RV062
     1*CQVAR(K)   +VDISP(I)/TVMIN(K)*FQVAR(K)                             RV063
 9972 GXV(S)*VDISP(J)*SDVAR(K,I)=QDVAR(K,I+1)/TVMIN(K)+VDISP(I)/TVMIN(K)  RV064
     1*FQVAR(K)-VPARI(I)/TVMIN(K)*GQVAR(K)                                RV065
      CXV(S)*DER1(QNVAR(K,J))=-VPARI(J)*PCOV(K,J)/TVMIN(K)*GQVAR(K)   *UD RV066
     1TXV(K)+VDISP(J)*PCOV(K,J)/TVMIN(K)*FQVAR(K)*UDTXV(K)                RV067
      GXV(S)*VPARI(J)*DER1(QPVAR(K,J))=-PCOV(K,J)/TVMIN(K)*GQVAR(K)   *UD RV068
     1TXV(K)                                                              RV069
 9973 GXV(S)*VDISP(J)*DER1(QDVAR(K,J))=+PCOV(K,J)/TVMIN(K)*FQVAR(K)*UDTX  RV070
     1V(K)                                                                RV071
```

Fig. 5 Listing of the APACHE subroutine for variable time delays.



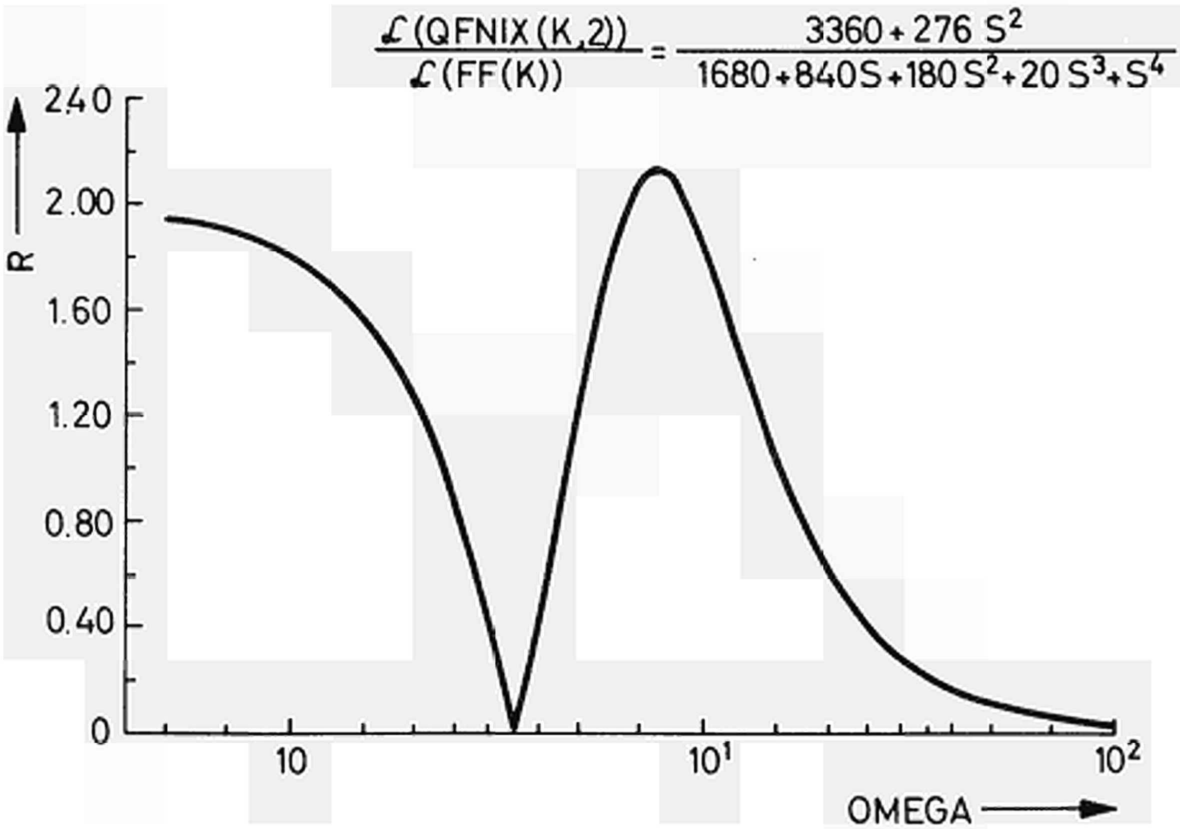
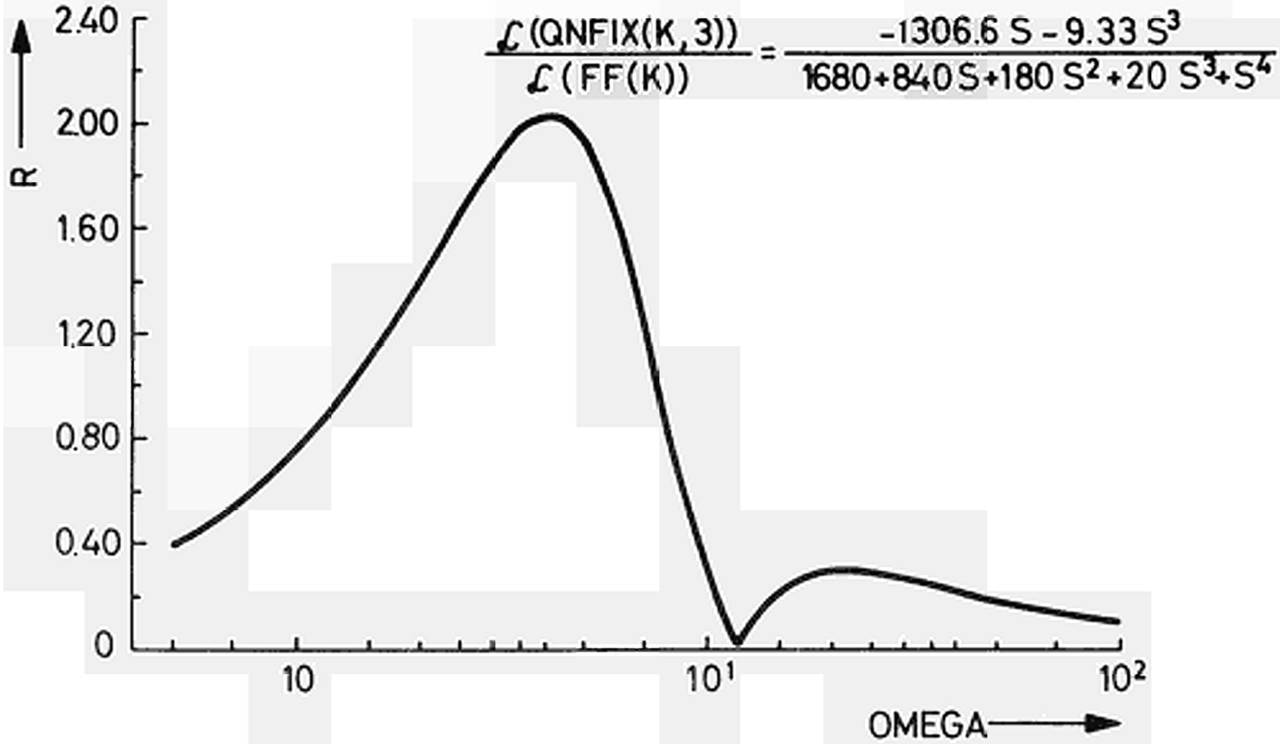


Fig. 6 Amplitude vs frequency diagram of $\frac{\text{QNFIX(K,1)}}{\text{FF(K)}}$ and $\frac{\text{QNFIX(K,2)}}{\text{FF(K)}}$.



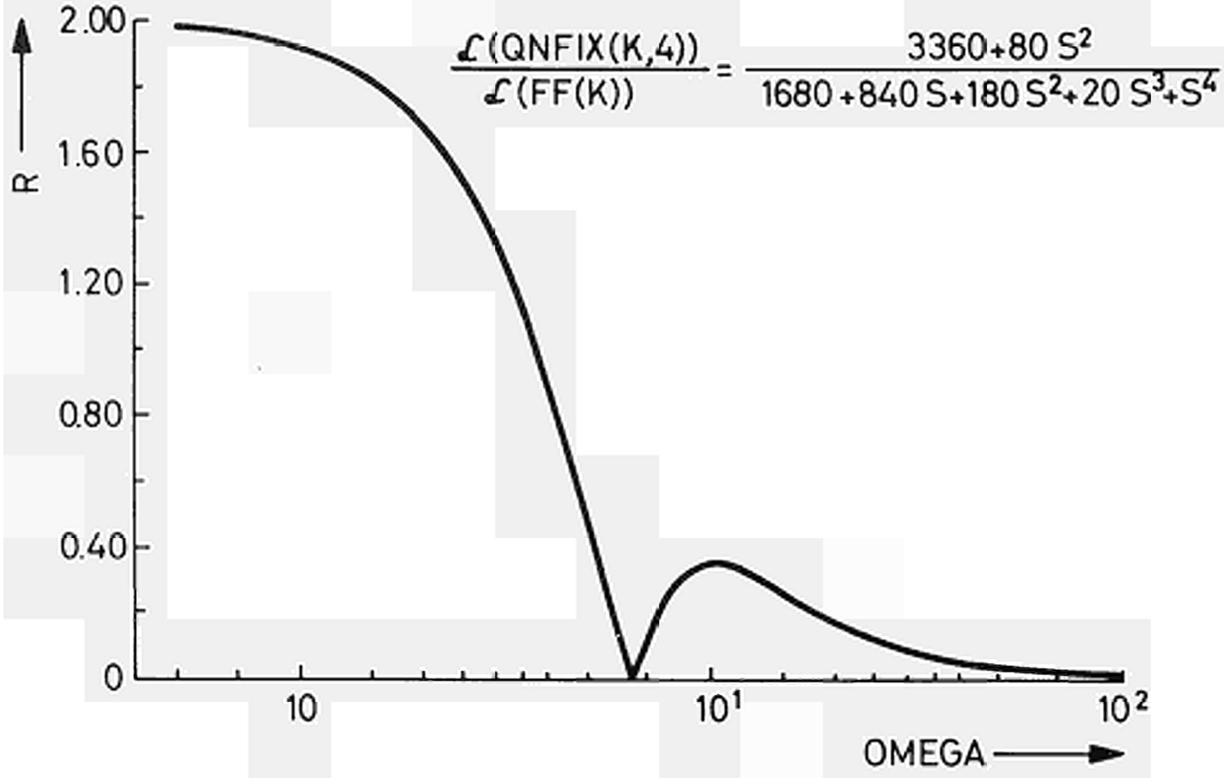


Fig. 7 Amplitude vs frequency diagram of $\frac{QNFIX(K,3)}{FF(K)}$ and $\frac{QNFIX(K,4)}{FF(K)}$.

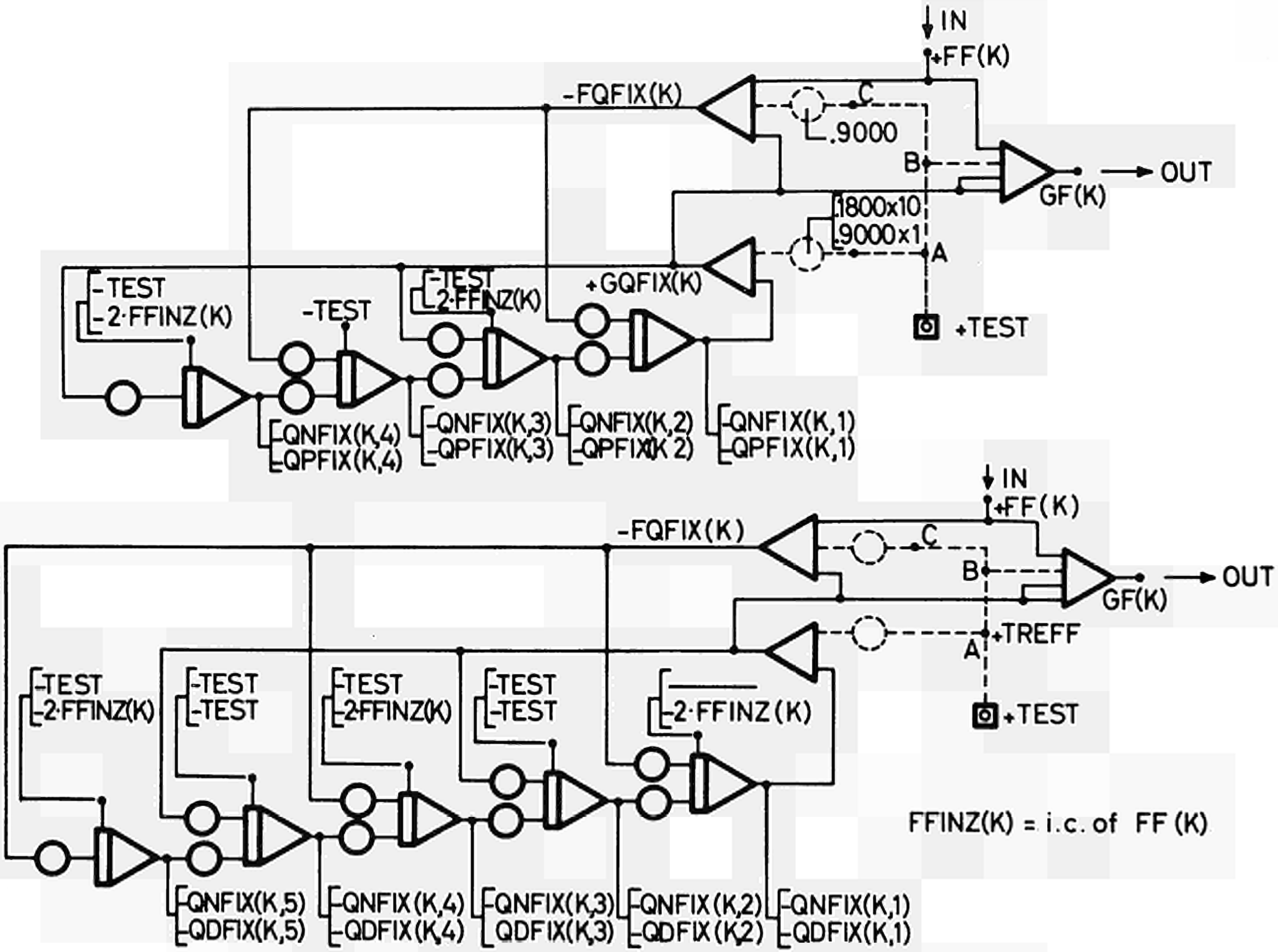


Fig. 8 Fixed time delay Pade circuits as deduced from the APACHE output listing.

Bibliography

1) EUR 2437.e (Euratom Report) 1965

APACHE : Analog Programming And CHEcking programmers manual.

2) EUR 3052.e (Euratom Report) 1966

APACHE : Analog Programming And CHEcking system programmers' guide.

3) EUR 240.f (Euratom Report) 1963

J.GAMP "Etude systématique de la génération de retards au moyen d'unités analogiques classiques".

4) EUR 347.f (Euratom Report) 1963

J. GAMP "Sous-routine APACHE pour l'introduction automatique des fonctions-retard fixes et variables dans les problèmes analogiques".

5) C.BONA How to use: "Two APACHE programs for the computation of circuits approximating a fixed or variable time delay according to the Padé scheme".

(To be published)

NOTICE TO THE READER

All Euratom reports are announced, as and when they are issued, in the monthly periodical **EURATOM INFORMATION,** edited by the Centre for Information and Documentation (CID). For subscription (1 year : US$ 15, £ 5.7) or free specimen copies please write to :

Handelsblatt GmbH
"Euratom Information"
Postfach 1102
D-4 Düsseldorf (Germany)

or

Office central de vente des publications
des Communautés européennes
2, Place de Metz
Luxembourg

To disseminate knowledge is to disseminate prosperity — I mean general prosperity and not individual riches — and with prosperity disappears the greater part of the evil which is our heritage from darker times.

Alfred Nobel

SALES OFFICES

All Euratom reports are on sale at the offices listed below, at the prices given on the back of the front cover (when ordering, specify clearly the EUR number and the title of the report, which are shown on the front cover).

OFFICE CENTRAL DE VENTE DES PUBLICATIONS DES COMMUNAUTES EUROPEENNES
2, place de Metz, Luxembourg (Compte chèque postal Nº 191-90)

BELGIQUE — BELGIË
MONITEUR BELGE
40-42, rue de Louvain - Bruxelles
BELGISCH STAATSBLAD
Leuvenseweg 40-42, - Brussel

DEUTSCHLAND
BUNDESANZEIGER
Postfach - Köln 1

FRANCE
SERVICE DE VENTE EN FRANCE DES PUBLICATIONS DES COMMUNAUTES EUROPEENNES
26, rue Desaix - Paris 15e

ITALIA
LIBRERIA DELLO STATO
Piazza G. Verdi, 10 - Roma

LUXEMBOURG
OFFICE CENTRAL DE VENTE DES PUBLICATIONS DES COMMUNAUTES EUROPEENNES
9, rue Goethe - Luxembourg

NEDERLAND
STAATSDRUKKERIJ
Christoffel Plantijnstraat - Den Haag

UNITED KINGDOM
H. M. STATIONERY OFFICE
P. O. Box 569 - London S.E.1

EURATOM — C.I.D.
51-53, rue Belliard
Bruxelles (Belgique)

CDNA03559ENC